THE FLORIDA STATE UNIVERSITY

COLLEGE OF ARTS AND SCIENCES

PPDA: PRIVACY PRESERVING DATA AGGREGATION IN WIRELESS SENSOR NETWORKS

By

KHANDYS A. POLITE

A Thesis submitted to the
Department of Computer Science
in partial fulfillment of the
requirements for the degree of
Master of Science

Degree Awarded:
Spring Semester, 2004

The members of the Committee approve the thesis of Khandys A. Polite defended on April 12, 2004.

Alec Yasinsac
Professor Directing Thesis

Mike Burmester
Committee Member

Greg Riccardi
Committee Member

Approved:

Sudhir Aggarwal, Chair, Department of Computer Science

The Office of Graduate Studies has verified and approved the above named committee members.

I dedicate this to my late cousin, Darrell Anson Young. Please know that you are always in my heart and on my mind. I love you and miss you terribly. Wish you were here.

XOXOXO

ACKNOWLEDGMENTS

I thank the Lord above first and foremost. Without Him, none of this would have been possible. I thank my mom, my sister, the rest of my magnificent family and all of my closest and dearest friends. Without their unconditional love and support and continuous words of encouragement and motivation I would not be the person I am today nor would I have achieved so much. I thank my major professor for directing me on this two-year journey towards achieving my master's degree. It has been a rough ride but well worth it. I thank my fellow colleagues who have helped me along the way. Some by challenging me to think about ideas and possibilities, some by helping me to relieve the stress with a few hardy laughs and some just by being there. Last, but certainly not least, I thank the Department of Defense (DoD), National Security Agency[1] (NSA) and Naval Research Laboratory (NRL) for giving me the Information Assurance Scholarship Program (IASP) scholarship and providing me with this absolutely amazing and wonderful opportunity. I was blessed with full and complete funding for the two years I pursued my master's degree so that I could be a full-time student and did not have to work, a summer internship in Washington, D.C. and now, a job guaranteed as soon as I graduate. I could not have planned it better myself. Thank you so much to everyone who helped me get where I am today. Much love to each and every one of you and may God bless you all.

[1] This material is based on work supported by the NSA under DoD Grant No. MDA904-03-10205.

TABLE OF CONTENTS

LIST OF FIGURES..........vii

ABSTRACT..........viii

1. INTRODUCTION..........1

1.1. Wireless Sensor Networks Described..........2
1.1.1. Data Delivery Models..........2
1.1.2. Network Routing Models..........3
1.1.3. Communication Approaches and Categories..........4
1.1.4. Thesis Structure..........4

2. SECURITY OF WIRELESS SENSOR NETWORKS..........5

2.1. Security Risks..........5
2.1.1. Intruder Sensor Devices..........5
2.1.2. Compromised Sensor Devices..........6
2.1.3. Eavesdroppers..........7
2.2. Security Requirements..........7
2.2.1. Data Integrity..........7
2.2.2. Data Authentication..........8
2.2.3. Data Confidentiality..........8
2.2.4. Data Freshness..........9

3. DATA AGGREGATION IN WIRELESS SENSOR NETWORKS..........10

3.1. Data Aggregation and Security..........10
3.2. Advantages of Data Aggregation..........11
3.3. Disadvantages of Data Aggregation..........11

4. RELATED WORK 13

4.1. SPINS 13
4.1.1. µTESLA 13
4.2. Secure Aggregation 16
4.2.1. Delayed Authentication 16
4.2.2. Delayed Aggregation 17
4.3. Technique for Remote Authentication 18

5. PRIVACY PRESERVING DATA AGGREGATION 21

5.1. Laying the Ground Work for PPDA 21
5.2. Composing Encryption and Aggregation Functions 27
5.3. Application of PPDA 29
5.3.1. Assumptions 29
5.3.2. Notation 31
5.3.3. Layout of Sensor Network 31
5.3.4. Key Generation 31
5.3.5. Data Communication Between Sensors and Base Station 32
5.3.6. Data Authentication, Encryption and Aggregation 33
5.3.7. A Simple Example 33

6. CONCLUSION 36

6.1. Future Work 36

REFERENCES 38

BIOGRAPHICAL SKETCH 40

LIST OF FIGURES

Figure 1 MAC Key Chain Generation and Authentication 14

Figure 2 MAC Key Disclosure 15

Figure 3 Delayed Data Aggregation.. 17

Figure 4 Remote Authentication Protocol 19

Figure 5 Remote Authentication Example 20

Figure 6 Example Sensor Network [2].. 22

Figure 7 Function Composition Equations 24

Figure 8 Generation of Signed Aggregate 24

Figure 9 Verification Process.. 25

Figure 10 Secure Generation of Signed Aggregate.. 26

Figure 11 Secure Verification Process 27

Figure 12 Generation of Encrypted Aggregate.. 28

ABSTRACT

Wireless sensor networks are undoubtedly one of the largest growing types of networks today. Much research has been done to make these networks operate more efficiently including the application of data aggregation. Recently, more research has been done on the security of wireless sensor networks using data aggregation. In this thesis, we discuss a method in which data aggregation can be performed securely by allowing a sensor network to aggregate encrypted data without first decrypting it.

CHAPTER 1

INTRODUCTION

Wireless sensor networks are fast becoming one of the largest growing types of networks today and, as such, have attracted quite a bit of research interest. They are used in many aspects of our lives including environmental analysis and monitoring, battlefield surveillance and management, emergency response, medical monitoring and inventory management [3, 2, 4]. Their reliability, cost-effectiveness, ease of deployment and ability to operate in an unattended environment [8, 4], among other positive characteristics, make sensor networks the leading choice of networks for these applications.

Data aggregation is an important mechanism used in sensor networks to conserve the already limited battery power of the sensor devices. Traditional approaches to data aggregation, which centered on the base station, required that all data be routed from sensor to sensor towards the base station where the aggregate of the data is calculated. This causes sensors to expend much of their energy transmitting data. More recently, research on data aggregation in sensor networks has resulted in energy-conserving and less centralized approaches in which data is aggregated at each sensor and the resulting aggregates are forwarded to the base station.

A major requirement of a sensor network is that the observer must be able to trust the result produced by the network. Clearly, this must also apply to networks using data aggregation. Colluding sensors that successfully mislead the observer into believing a false result about the network are a serious threat and thus must be protected against. The focus of this thesis is on providing a secure data aggregation mechanism that protects against collusion attacks in a wireless sensor network using distributive properties of aggregation and encryption function composition.

1.1. Wireless Sensor Networks Described

The primary function of a wireless sensor network is to determine the state of the environment being monitored by sensing some physical event. Wireless sensor networks consist of hundreds or thousands or, in some cases, even millions of sensor devices that have limited amounts of processing power, computational abilities and memory and are linked together through some wireless transmission medium such as radio and infrared media. These sensors are equipped with sensing and data collection capabilities and are responsible for collecting and transmitting data back to the observer of the event.

Sensors may be distributed randomly in an ad hoc manner and may be installed in fixed locations or they may be mobile. For example, they may be deployed by dropping them from an aircraft as it flies over the environment to be monitored. Once distributed, they may either remain in the locations in which they landed or they may begin to move if necessary. Sensor networks are dynamic because of the addition and removal of sensors due to device failure in addition to mobility issues.

1.1.1. Data Delivery Models

The authors of [8] discuss four data delivery models used to classify sensor networks based on the requirements of the observer: continuous, event-driven, observer-initiated and hybrid. In the continuous model, sensors use a predetermined rate by which to transmit their data continually. For example, a sensor may be required to transmit a temperature reading every five minutes. In the event-driven model, sensors will only transmit data if an event of interest takes place. For example, a sensor placed at a traffic light may be required to transmit data every time a vehicle runs a red light. In the observer-initiated model, sensors will only transmit data when the observer issues a request. The observer will typically submit a query or an interest about an event to the network through some intermediate medium, like a sink [10, 6] or a base station [3, 2][2]. For example, an observer may request location information from a sensor armed with a location finding application. In the hybrid model, the first three models coexist in the

[2] Throughout the remainder of this thesis, we will use the term base station as opposed to sink.

same network. Consider animal habitat monitoring as an example. Temperature readings can be transmitted on a continuous basis, location information can be transmitted each time an animal moves and the observer can submit a request to determine the number of animals currently being monitored.

1.1.2. Network Routing Models

The ad hoc nature of sensor networks implies an infrastructureless network. As a result of this, once the sensors have been deployed, they must organize themselves into a functional network. Sensors must communicate amongst themselves to establish routing paths through which data will be transmitted between the observer and the event. New routing paths must be established whenever sensors suffer from device failure or whenever new sensors are added to the network. Several "self-configuring" protocols and algorithms have been designed for wireless sensor networks including the Self-Organizing Medium Access Control for Sensor Networks (SMACS) protocol [9] and Eavesdrop-And-Register (EAR) algorithm [9].

The authors of [8] speak of two network routing models used to classify sensor networks based on the mobility aspect of the network: static sensor networks and mobile sensor networks. In the static model, the sensors, observer and event are all stationary. There is no movement among them. Upon initialization, static sensor networks only need establish the communication infrastructure once. This creates the routing paths. Note, however, that additional infrastructure communication will be necessary in order to maintain this path whenever a sensor device fails.

In the mobile model, the sensors themselves, the observer, or the event are mobile. At any time, a path might fail. A new path may be established either through the observer-initiated approach or the sensor-initiated approach. The observer-initiated approach to establishing a new path is only used if the observer notices a broken path. The sensor-initiated approach is used whenever a sensor anticipates breaking the path by moving out of range.

1.1.3. Communication Approaches and Categories

Communication approaches are different from data delivery models. Data delivery models are based on the requirements of the observer and, in essence, determine the method of sensing used by the sensors. Communication approaches define how the data actually flows between the sensors and the observer. The authors of [8] address three communication approaches: flooding, unicast and multicast/other.

Flooding is a form of broadcasting in which sensors broadcast their data to neighboring sensors. When a sensor receives broadcasted data, it rebroadcasts the data to its neighbors. This continues until the data finally reaches the observer. In the unicast approach, sensors can transmit data either directly to the observer using a multi-hop routing protocol or, if clustering is applicable, they can transmit the data to the cluster-head using one-to-one unicast. In the multicast approach, sensors form application-directed groups and use multicasting to transmit data back and forth between group members.

1.1.4. Thesis Structure

The remainder of this thesis will be organized as follows. Chapter 2 discusses the security of wireless sensor networks. Chapter 3 discusses data aggregation as a way to conserve energy in wireless sensor networks. Chapter 4 provides background information on work related to this thesis. Chapter 5 describes in detail our privacy preserving data aggregation technique. This thesis is then concluded in Chapter 6.

CHAPTER 2

SECURITY OF WIRELESS SENSOR NETWORKS

Security in wireless sensor networks is a major challenge. The limited amount of processing power, computational abilities and memory with which each sensor device is equipped makes security a difficult problem to solve. Traditional security techniques and algorithms such as asymmetric key cryptography and public key cryptography are infeasible for sensor networks because they require large amounts of memory for long messages, keys and digital signatures and large amounts of power for expensive and battery-draining computations [3, 2]. Instead, security mechanisms such as symmetric key cryptography are routinely the tools of choice. Care must still be taken when developing a secure protocol using symmetric key mechanisms as they may deplete the already constrained resources of the sensor devices.

2.1. Security Risks

Sensor networks are vulnerable to insider and outsider attacks just like any other wireless network. When designing a security protocol or algorithm it is important to understand the dangerous and damaging effects these attacks can have so that the protocol or algorithm can guard against them. To understand the attacks, one must be aware of the security risks of the network. We consider three categories of security risks in sensor networks: intruder sensor devices, compromised sensor devices and eavesdroppers.

2.1.1. Intruder Sensor Devices

Without a secure method of transmitting data in sensor networks, i.e. secure routing protocol and/or message authentication, an adversary can easily deploy his own sensor devices and inject false data into the network. A non-secure network routing protocol used by the sensors to build the routing paths will allow an intruder sensor to include itself in routing paths. A non-secure communication approach to transmitting data will allow an intruder sensor to transmit false data.

One approach to protecting against intruder sensors is for each sensor to be initialized with a unique identifier and secret key before it is deployed. The identifier serves to establish the device as a legitimate and identifiable part of the network. The secret key is used for data authentication[3]. Many of the security protocols currently in existence for sensor networks employ these methods as a way to guard against intruder sensor devices including [1], [2] and [3].

In [1], the base station maintains a list of all sensors in the network by storing their identifiers. Data being transmitted by a sensor is encrypted using an encryption key that is generated by XORing a session key with the sensor's secret key. The sensor then appends its identifier and a time stamp to the encrypted data before sending it. The base station, knowing the secret key of the sensor, computes the sensor's encryption key and decrypts the data. This method provides authentication through privacy using the secret encryption key of a sensor and thus protects against intruder sensors.

In [2] and [3], the identifier of the sensor is also appended to the data to be transmitted. Sensors then authenticate the data after the base station releases the secret authentication key. This method is further explained in chapter 4.

2.1.2. Compromised Sensor Devices

We consider compromised sensor devices to be the most dangerous threat. A compromised sensor is an authorized sensor that has been captured by an adversary, possibly by some physical means of tampering with the device. An adversary having a compromised sensor in his possession can inject false data or modify, forge, or discard data received from another sensor without being detected because he has access to the identifier and secret key (if the key has also been compromised) that allowed the sensor to be a valid part of the network.

[3] It can also be used for data encryption, which will be discussed later, but authentication is most important with respect to intruder sensors.

It is difficult to prevent a sensor from being compromised so some protocols attempt to limit the amount of damage a compromised sensor can do rather than attempting to prevent it completely [2, 3]. The solutions provided by these protocols will be discussed in chapter 4.

2.1.3. Eavesdroppers

There are two categories of eavesdroppers: inside eavesdroppers and outside eavesdroppers. Outside eavesdroppers can be adversaries outside the network who are able to receive data transmissions in the network. Inside eavesdroppers can be intruder or compromised sensors that exist inside the network and are able to receive data transmissions not meant for them. The danger from eavesdropping is that sensitive data may be leaked to an unauthorized entity. Both inside eavesdroppers that are intruder sensors and outside eavesdroppers can be deterred through data encryption.

2.2. Security Requirements

The security of a sensor network is dependant upon four requirements that must be fulfilled: the integrity of the data must be ensured, the sender of the data must be verified and the data authenticated, the data must not be leaked to an unauthorized source and the data must be recent [3].

2.2.1. Data Integrity

Data integrity guarantees that the data has not been altered, forged or tampered with in an unauthorized way as it is transmitted from sender to receiver. The receiver may be one or several hops away from the sensor sending the data. This means that the data may need to be forwarded to the receiver from the sensor via intermediate sensors.

Methods of ensuring data integrity include controlling the environment, restricting access to data and using authentication techniques. It may not be feasible to use environment control as a method because sensors may be deployed in hostile, unattended environments. Restricting access to data is typically used in networks where the security levels of the data (classified, secret, top secret) differ and may not be applicable to sensor networks. Authentication is quite often the method of choice. In addition to providing integrity, it can also provide data origin verification.

2.2.2. Data Authentication

Data authentication allows the receiver to verify the origin of received data and ensure that it originated from a trusted source and not from an intruder or compromised sensor. It is accomplished through the use of a Message Authentication Code (MAC). Hashing the data with a keyed hash function where the key used is a symmetric shared key generates a MAC. Typically, the data and MAC are transmitted together. The receiver generates a MAC on the data it receives using the symmetric shared key and compares this to the MAC it received with the data. If the two match, then it is concluded that the data came from a trusted source. The use of a shared symmetric key is what proves to the receiver that the sender of the data is a trusted source. If the two generated MACs do not match, then it is possible that the sender of the data used a different key and thus may not be trusted.

As was previously mentioned, data authentication also provides data integrity. If the two generated MACs do not match, then the data has been altered in transmission.

2.2.3. Data Confidentiality

Data confidentiality keeps data secret from both inside and outside eavesdroppers. This is necessary in networks where sensitive data is transmitted. Battlefield surveillance is one example. Highly sensitive data about the enemy may be transmitted within the sensor network and, if confidentiality is not addressed, the network runs the risk of leaking sensed data to other networks, possibly enemy networks.

The traditional approach to providing confidentiality of data is to use data encryption. Only authorized sensors are able to decrypt the data, thus keeping it secret from unauthorized eavesdroppers.

2.2.4. Data Freshness

Data freshness provides protection against replay attacks. This is important in a sensor network because the usefulness of the network depends on the trustworthiness of the data. An adversary who replays old information is essentially the same as an intruder or compromised sensor that alters or forges data. Ultimately, the adversary misleads the observer about the state of the environment being monitored.

Data freshness can be achieved through the use of counters, timers and/or one-time use keys. For example, a counter may be concatenated to the data before a MAC is generated on it as in [3] or, a counter can be encrypted using a shared symmetric key to obtain a temporary MAC key or encryption key as in [2].

CHAPTER 3

DATA AGGREGATION IN WIRELESS SENSOR NETWORKS

Due to the power constraints of the sensor devices, it is necessary to conserve energy. The authors of [9] partition energy consumption into three categories: sensing, data processing, and communications. The communications category consumes the most energy of the three [3, 9, 7]. In order to reduce the overhead of data communications and thereby conserve energy, a sensor network must reduce the amount of data that is transmitted. Data aggregation, in which sensors combine data, contributes to this reduction.

3.1. Data Aggregation and Security

Data aggregation must be done securely so as to prevent a deceptive reading of the state of the environment being monitored. If an attacker is successful in misleading the observer, then the network is useless because it does not accomplish the task it was employed to do. For this reason, a sensor network that uses data aggregation is required to protect the integrity of the aggregated data just as it was required to protect the integrity of the non-aggregated data. A sensor that receives data from other sensors and performs aggregation must not be able to substitute an incorrect aggregate value. It must calculate the aggregate using the data received from the other sensors.

It is also essential to prevent leakage of sensitive data. If, for example, a network were responsible for monitoring a military target for the purpose of planning a surprise attack, then it would be necessary to ensure that the privacy of the information is preserved so that the target does not become aware of the ensuing plans. For this reason, a sensor network that uses data aggregation is also required to protect the confidentiality of the aggregated data just as it was required to protect the confidentiality of the non-aggregated data.

3.2. Advantages of Data Aggregation

In a sensor network, data is routed from the sensors to the observer via the base station. Along the way, redundant and irrelevant data is encountered. By reducing the redundancy and only transmitting the relevant data, the number of data transmissions can be reduced. This, in turn, reduces the amount of energy expended in relaying the data. Data aggregation, an in-network data processing mechanism, is used to achieve this [10, 6, 5, 13].

Data aggregation may also be used to summarize data in such a way as to relay relevant and necessary information without requiring that all pieces of data be transmitted. In this case, data may be aggregated using functions to compute averages, sums, median values, minimum and maximum values, etc. [12, 14]. For example, an observer may be interested in obtaining the average temperature reading of the monitored environment. Rather than having every sensor transmit individual temperature readings to the base station where the average would be calculated before being sent to the observer, the average is calculated as the data flows towards the base station.

3.3. Disadvantages of Data Aggregation

Data aggregation complicates the security of sensor networks [2]. As was previously discussed, compromised sensor devices are the most dangerous security risks because they have the ability to completely misrepresent the state of the environment and mislead the observer by transmitting false data (including aggregate data). A compromised sensor may receive legitimate data from other sensors but may modify the data before aggregation. Transmitting the falsified aggregate may allow the compromised sensor to misrepresent sensors from which data was received and aggregated.

Data encryption is essential to providing the data confidentiality in sensor networks. However, aggregation makes data encryption more difficult to accomplish [2]. Often, sensors need to understand the data that they receive from other sensors in order to aggregate it. This means that they must be able to decrypt the encrypted data before they can proceed with the aggregation. The base station must also decrypt the encrypted data in order to relay the aggregated information to the observer. This implies that a key-sharing scheme of is necessary.

Asymmetric key cryptosystems are typically not well suited to sensor networks because they are computationally intensive. This intensity absorbs much of the power and computational abilities of the sensor devices and it is for this reason that symmetric keys are used. However, there are problems with this solution, too. Having the base station share a different unique key with each sensor will not allow intermediate sensors to decrypt the data they receive. Only the base station and the sensor encrypting the data will be able to decrypt it. Having the base station and all the sensors share the same unique key will allow intermediate sensors to decrypt the data but will also allow a compromised sensor to control most, if not all, of the network. In a fully connected sensor network, a compromised sensor in this scenario can effectively masquerade as any sensor in the network or, worse yet, the base station itself.

An ideal solution is to use symmetric keys in an asymmetric fashion to combine the advantages of the two key-sharing cryptosystems. The protocols discussed in [2] and [3] employ this idea. They achieve asymmetry using symmetric keys and a time delay. This is discussed in greater detail in chapter 4.

CHAPTER 4

RELATED WORK

There are three main works related to this thesis: the SPINS protocol [3], the Hu and Evans protocol[4] [2] and the Wulf, Yasinsac, Oliver and Peri technique[5] [11].

4.1. SPINS

SPINS [3] is a suite of security protocols that is claimed to satisfy the four security requirements: data integrity, data authentication, data confidentiality and data freshness. It consists of two building blocks: the Security Network Encryption Protocol (SNEP) and a micro version of Timed Efficient Stream Loss-tolerant Authentication (TESLA) called µTESLA. For the purposes of this thesis, it is not necessary to discuss the SNEP in detail. It is only necessary to discuss µTESLA.

4.1.1. µTESLA

The base station uses authenticated broadcast to transmit data to the sensor devices. Authenticated broadcast is best achieved through the use of asymmetric cryptography but can also be achieved through the use of symmetric cryptography. Traditional asymmetric mechanisms require large amounts of storage and expensive computations neither of which are possibilities with a resource-constrained sensor device. Instead, µTESLA provides authenticated broadcast using symmetric cryptography and asymmetry is obtained via a technique called delayed key disclosure.

[4] For the remainder of this thesis, this protocol will be referred to as the Hu-Evans protocol.
[5] For the remainder of this thesis, this protocol will be referred to as the WYOP technique.

Delayed key disclosure. Delayed key disclosure is based on time intervals and symmetric keys, which are established by the sender, in this case, the base station. The keys are used by the base station to generate MACs on the data to be sent and by the receiver, the sensor, to authenticate the data received. To obtain the keys, the base station generates what is known as a key chain of secret MAC keys. To begin, the base station chooses the last key in the key chain, K_n, randomly. The next key in the key chain, K_{n-1}, is generated by applying a public one-way function, F, to the previous key, K_n, Figure 1a.

K_n = last key in chain	K_4 = last key in chain	$K_0 = F(K_1)$
$K_{n-1} = F(K_n)$	$K_3 = F(K_4)$	$K_1 = F(K_2)$
$K_{n-2} = F(K_{n-1})$	$K_2 = F(K_3)$	$K_2 = F(K_3)$
$K_{n-3} = F(K_{n-2})$	$K_1 = F(K_2)$	$K_3 = F(K_4)$
$K_{n-4} = F(K_{n-3})$	$K_0 = F(K_1)$	K_4 = last key in chain
$\vdots$	$\boxed{\text{Ex. } n = 4}$	
(a)	(b)	(c)

Figure 1 MAC Key Chain Generation and Authentication

Applying F to the previously generated key generates each successive key. The resulting key chain contains n + 1 keys. As an example, choose n = 4, Figure 1b. The resulting key chain contains 5 keys, K_0, K_1, K_2, K_3 and K_4. Once the key chain has been generated, each MAC key in the chain is assigned to a time interval so that each MAC key will only be used during its assigned time interval.

In order for the sensors to authenticate the received data, the secret MAC key must be revealed to them. MAC keys are revealed periodically on a time schedule set by what is known as the key disclosure time delay, δ. The key disclosure time delay is an arbitrary number of time intervals after which the MAC key is revealed. If MAC key K_i is used by the base station in time interval i, then K_i is revealed in time interval i + δ. Once the sensors receive the MAC key, the authenticity of the key must be verified before the data can be authenticated. To achieve key authentication, the sensors apply the public one-way function to the received key and compare the result to the previously received and authenticated key. If the two match, then the currently received key is deemed authentic and the data can be authenticated. For example, if a sensor receives MAC key K2, it verifies the authenticity of it by applying F to it, Figure 1c. The result should match the previously received and authenticated key, K_1.

As an example of delayed key disclosure, observe Figure 2.

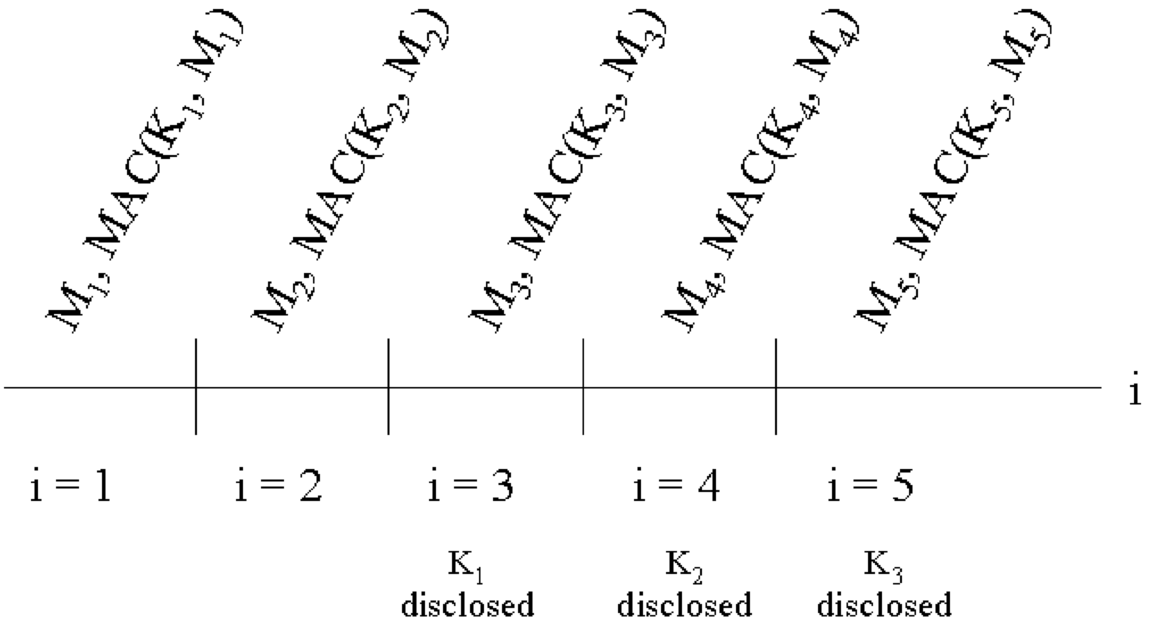


Figure 2 MAC Key Disclosure

Time has been divided into five intervals, i. Key K_1 is the MAC key used in time interval 1, key K_2 is the MAC key used in time interval 2 and so on. Message M_1 is a message that will be transmitted during time interval 1. Although the figure shows one message being transmitted per time interval, it is feasible to transmit more than one message in the same time interval using the same MAC key. If the time delay is 2 time intervals, then MAC key K_1 will be revealed during time interval 3, MAC key K_2 will be revealed during time interval 4 and so on.

4.2. Secure Aggregation

The Hu-Evans protocol [2] is a protocol that satisfies three of the four security requirements: data authentication, data freshness and data integrity. The protocol uses two main ideas: delayed authentication (inspired mostly by SPINS) and delayed aggregation. Using concepts from SPINS, the protocol provides secure data aggregation in sensor networks.

4.2.1. Delayed Authentication

In the Hu-Evans protocol, μTESLA is used to achieve delayed authentication for data transmitted by the base station, which uses authenticated broadcast to transmit data to the sensor devices. It is also necessary for sensor devices to transmit data that are authenticated by other sensors and the base station. For this, each sensor generates its own set of MAC keys one at a time (as opposed to all at once, like the key chain the base station generates). To produce a key, the sensor encrypts a counter value using a secret key that it shares with the base station. The counter value used and the MAC key generated are both tied to the time interval in which the data is transmitted and so are used only during that interval.

Broadcasting data is an expensive task for sensors so, to conserve energy, the base station reveals the MAC key to the network on behalf of the sensors. For this to work, it is necessary for the base station and the sensors to synchronize counter values. The base station calculates the MAC key the same way the sensor did by encrypting the counter value using the secret key it shares with the sensor. The MAC key is revealed after the key disclosure time delay as it is in µTESLA and the sensors and base station then authenticate the data.

4.2.2. Delayed Aggregation

Delayed data aggregation is included in the Hu-Evans protocol as a means of minimizing the amount of damage a compromised sensor can cause. Without it, a compromised sensor can tamper with data received from other sensors. The idea of delayed data aggregation is that, rather than having parents aggregate the data of their children (immediate, next-hop data aggregation), grandparents aggregate the data of their grandchildren (delayed, second-hop data aggregation).

As an example of delayed aggregation, observe Figure 3.

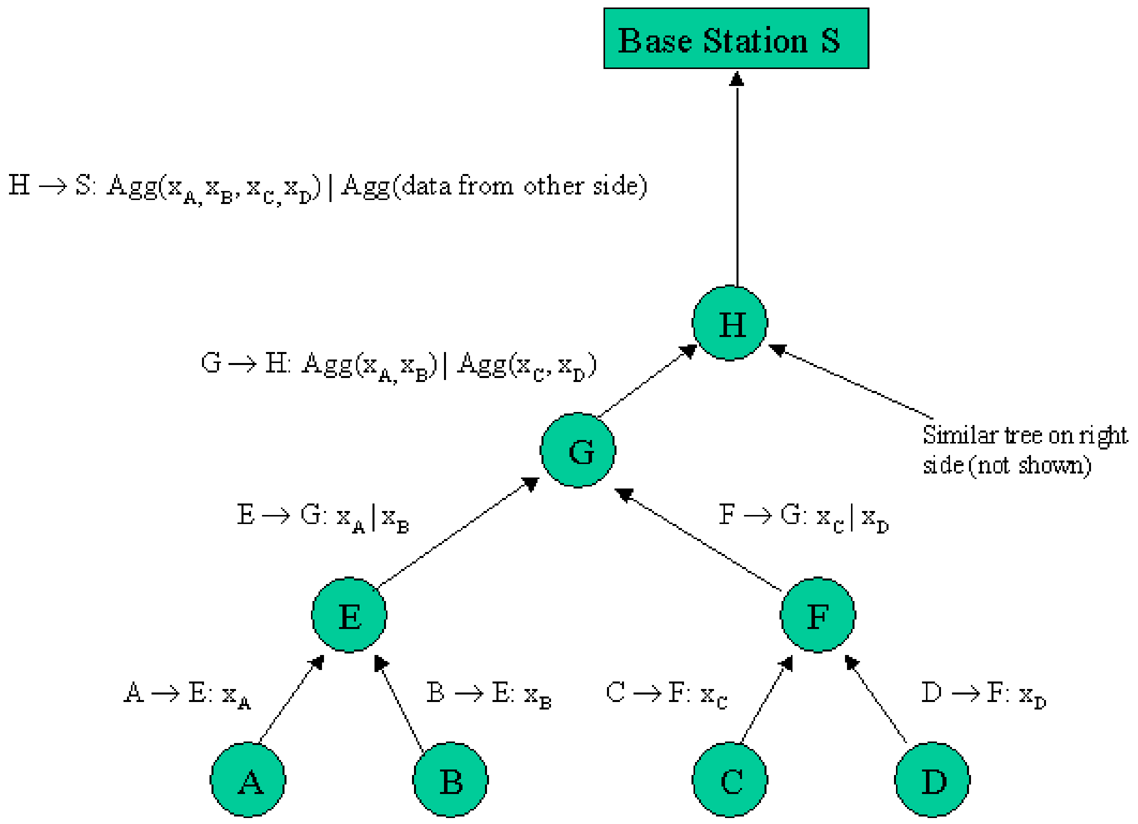

Base Station S
H → S: Agg(x_A, x_B, x_C, x_D) | Agg(data from other side)
H
G → H: Agg(x_A, x_B) | Agg(x_C, x_D)
G
Similar tree on right side (not shown)
E → G: x_A | x_B
F → G: x_C | x_D
E
F
A → E: x_A
B → E: x_B
C → F: x_C
D → F: x_D
A
B
C
D

Figure 3 Delayed Data Aggregation

Sensors A and B transmit their data, x_A and x_B, respectively, to their parent sensor, E. Sensors C and D transmit their data, x_C and x_D, respectively, to their parent sensor, F. Sensors E and F forward the data they received from their children and do not perform aggregation. Sensor G calculates and transmits two aggregates: an aggregate of the data received from sensors A and B through E and an aggregate of the data received from sensors C and D through F. The grandparent, G, aggregated the data of its grandchildren, A, B, C and D.

The authors of [2] address the fact that their protocol does not provide protection against parent-child sensor compromises and they also offer a solution. We discovered vulnerabilities in the protocol and in the solution offered and discuss these further in chapter 5. These discoveries led us to search for ideas to provide a better solution that would protect against collusion attacks. The work discussed in the next section gave us such an idea.

4.3. Technique for Remote Authentication

The WYOP technique [11] allows a participant to authenticate another participant without the need for prior shared knowledge. It incorporates characteristics of public key cryptography, zero knowledge proofs and composition of functions in a way that may effectively prevent collusion attacks.

The idea behind this technique is as follows. Find two functions, f and g, such that when you compose them

$$f(g(x,y)) = g(f(x), f(y)). \qquad (1)$$

This equality serves as the authentication of one participant to another. The participants exchange a series of values in the clear and perform calculations on the received values. In the end, the verifier compares the values $f(g(x,y))$ and $g(f(x), f(y))$ to be sure that they are equal.

Participant A selects function f secretly and participant B selects function g secretly. When A wishes to communicate with B, he generates a random value, x, calculates f(x) and sends those two values to B, Figure 4a.

$A \rightarrow B: x, f(x)$

$B \rightarrow A: y, g(x, y)$

$A \rightarrow B: f(y), f(g(x, y))$

(a)

A knows: $x, y, f(x), f(y), g(x, y), f(g(x, y))$

B knows: $x, y, f(x), f(y), g(x, y), f(g(x, y))$

B calculates: $g(f(x), f(y))$

B verifies: $f(g(x, y)) = g(f(x), f(y))$

(b)

Figure 4 Remote Authentication Protocol

B responds by generating a random value, y, calculates g(x,y) using the x value received from A and sends those two values to A. A then calculates f(y) and f(g(x,y)) using the y and g(x,y) values received from B and sends those two values to B. At this point, the exchange of information is finished and both A and B know the information in Figure 4b. B calculates g(f(x), f(y)) and compares this value to f(g(x,y)). If they are the same, then B can authenticate A.

The trick behind this is selecting two private, hard-to-invert functions where one of the functions (the function of the authenticating participant) distributes over the other. Selecting private functions is similar to selecting private keys. It is important to keep the functions secret to protect against a masquerading attack. Even though the functions are secret, the forms of the functions can be made public. Using hard to invert functions is similar to using hard to invert functions for generating encrypted data. It is important for the functions to be hard to invert because the data is exchanged in the clear. If the functions were not hard to invert, then an eavesdropper who is listening to the entire conversation and obtains the exchanged values would be able to use these values to determine the functions and later masquerade as one of the participants.

The distributive property of the functions is necessary so that the composition of the two functions produces equation (1) which is used for the authentication. Figure 5 shows an example exchange between two participants, A and B where A's function is $f(x) = x^a \bmod n$ and B's function is $g(x,y) = (xy)^b \bmod n$. These functions are hard to invert and f distributes over g because of the distributive property of exponentiation over multiplication.

(a)

$A \rightarrow B$: x, $x^a \bmod n$

$B \rightarrow A$: y, $(xy)^b \bmod n$

$A \rightarrow B$: $y^a \bmod n$, $((xy)^b \bmod n)^a \bmod n$

(b)

A knows: x, y, $x^a \bmod n$, $y^a \bmod n$, $(xy)^b \bmod n$, $((xy)^b \bmod n)^a \bmod n$

B knows: x, y, $x^a \bmod n$, $y^a \bmod n$, $(xy)^b \bmod n$, $((xy)^b \bmod n)^a \bmod n$

B calculates: $((x^a \bmod n)(y^a \bmod n))^b \bmod n = ((xy)^b \bmod n)^a \bmod n$

B verifies: $((xy)^b \bmod n)^a \bmod n = ((xy)^b \bmod n)^a \bmod n$

Figure 5 Remote Authentication Example

CHAPTER 5

PRIVACY PRESERVING DATA AGGREGATION

Privacy preserving data aggregation (PPDA) is a method by which a sensor network can securely aggregate data. PPDA maintains the confidentiality of the data as it is aggregated. Providing data confidentiality protects the data against inside and outside eavesdroppers and compromised sensors and ensures that sensitive information is not leaked to an adversary.

5.1. Laying the Ground Work for PPDA

The protocol described in [2] uses delayed data aggregation to ensure data integrity but only succeeds in ensuring the integrity of data in a network where two consecutive sensors are not compromised. This means that it does not secure against a parent-child sensor compromise. Consider the example sensor network of [2], Figure 6.

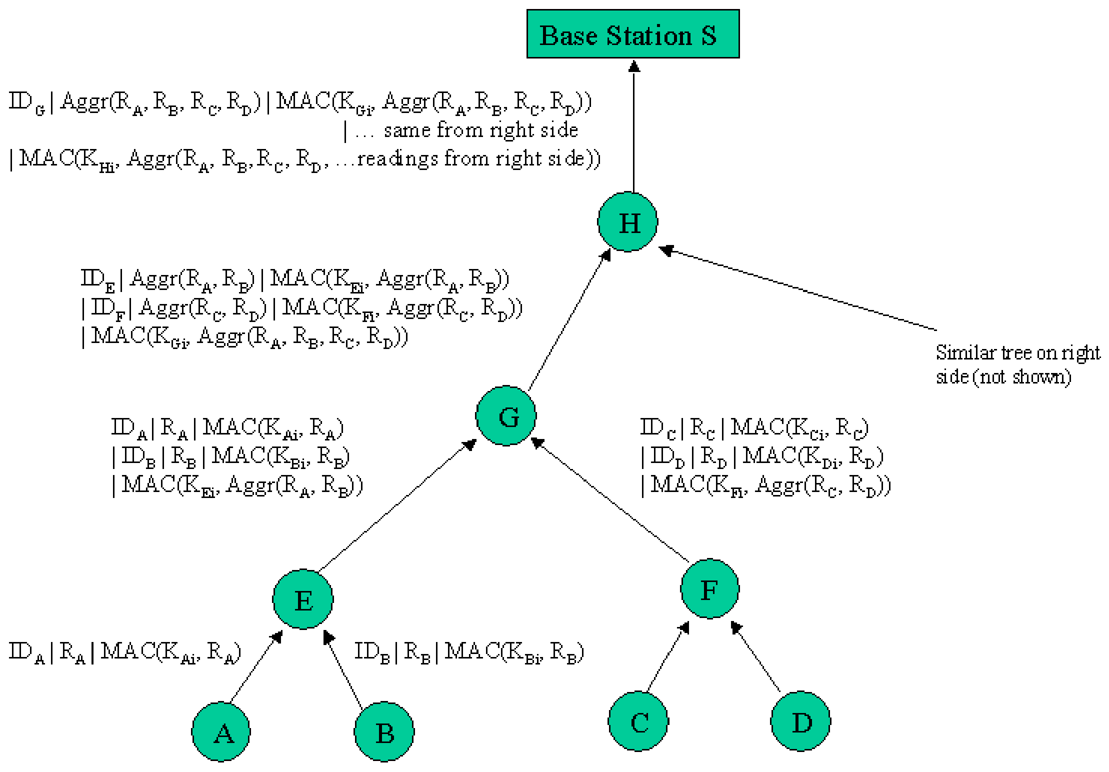

Base Station S
ID_G | Aggr(R_A, R_B, R_C, R_D) | MAC(K_{Gi}, Aggr(R_A, R_B, R_C, R_D))
| … same from right side
| MAC(K_{Hi}, Aggr(R_A, R_B, R_C, R_D, …readings from right side))
H
ID_E | Aggr(R_A, R_B) | MAC(K_{Ei}, Aggr(R_A, R_B))
| ID_F | Aggr(R_C, R_D) | MAC(K_{Fi}, Aggr(R_C, R_D))
| MAC(K_{Gi}, Aggr(R_A, R_B, R_C, R_D))
Similar tree on right side (not shown)
G
ID_A | R_A | MAC(K_{Ai}, R_A)
| ID_B | R_B | MAC(K_{Bi}, R_B)
| MAC(K_{Ei}, Aggr(R_A, R_B))
ID_C | R_C | MAC(K_{Ci}, R_C)
| ID_D | R_D | MAC(K_{Di}, R_D)
| MAC(K_{Fi}, Aggr(R_C, R_D))
E
F
ID_A | R_A | MAC(K_{Ai}, R_A)
ID_B | R_B | MAC(K_{Bi}, R_B)
A
B
C
D

Figure 6 Example Sensor Network [2]

If the parent and child sensors, G and E, respectively, are compromised, then they can collude to successfully modify or tamper with the data transmitted by sensors A and B. This is possible because the sensors E and G are the only sensors that verify the integrity of this data. Sensor H is only responsible for verifying the integrity of the aggregate of the data transmitted by sensors A and B.

The protocol in [2] also does not secure against grandparent-grandchild sensor compromises. In Figure 6, if sensors H and E are compromised, then they can also collude to successfully forge or discard the data transmitted by sensors A and B. The protocol looks to avoid this by requiring the parent of the compromised sensor to raise an alarm if it suspects sensor misbehavior, which is detected if the MAC received from E does not match the MAC generated by G in the authentication phase. However, the grandparent (H) of the compromised sensor (E) may choose to discard any and all data it receives from the parent (G), including the alarm message, and transmit its own data instead.

The solution proposed by [2] is to delay the data aggregation another level. One reason why this solution is not ideal is that it increases the transmission cost due to the increase in the amount of data that is transmitted. Another reason is that, no matter how many levels the aggregation is delayed, performing intermediate integrity checks as the data flows towards the base station still allows intermediate sensors to falsify data because there is always a level at which data from levels below is no longer being checked.

The optimal solution is to have the base station perform the integrity check because it is the only case where intermediate sensors cannot falsify data, either on their own or by colluding. This is typically accomplished by sending all raw data and MACs to the base station without any in-network processing. However, this consumes energy. To conserve energy and lower transmission costs, it is necessary to perform in-network data aggregation. This implies that if the base station is to be responsible for performing the integrity check, it must be done without transmitting the raw data to the base station. The base station needs to use the final aggregated value to perform the integrity check.

Using the idea of composition of functions as outlined in [11], we were interested in finding two functions that, when composed properly, would ensure data integrity and prevent collusion. Conceptually, one of the functions was to be a signature function and the other was to be an aggregation function. The idea was to intertwine the signing and aggregating of data as it flowed up the network towards the base station. Once the signed aggregate reached the base station, it would be verified.

Ultimately, we wanted the equation in Figure 7a to hold true. In other words, we wanted the signature of the aggregated data to match the aggregation of the signatures. For a network consisting of three levels of sensors (hierarchical routing), the equality would be as in Figure 7b. A couple of interesting patterns to notice are: 1) for every signature generated on the left-hand side, two signatures are generated on the right-hand side and 2) the number of aggregations is the same on both sides.

(a) $\text{sign}(\text{aggregate}(x, y)) = \text{aggregate}(\text{sign}(x), \text{sign}(y))$

(b) $\text{sign}(\text{agg}(\text{sign}(\text{agg}(x_1, y_1)), \text{sign}(\text{agg}(x_2, y_2)))) = \text{agg}(\text{sign}(\text{agg}(\text{sign}(x_1), \text{sign}(y_1))), \text{sign}(\text{agg}(\text{sign}(x_2), \text{sign}(y_2))))$

Note: *agg* is short for aggregate.

Figure 7 Function Composition Equations

Using the simple example in [11], we were able to show that the idea was possible. The signature function used was $f(x) = x^2$ and the aggregation function used was $g(x, y) = xy$. Each leaf sensor transmits its collected data, x, and a signature on that data, f(x), Figure 8.

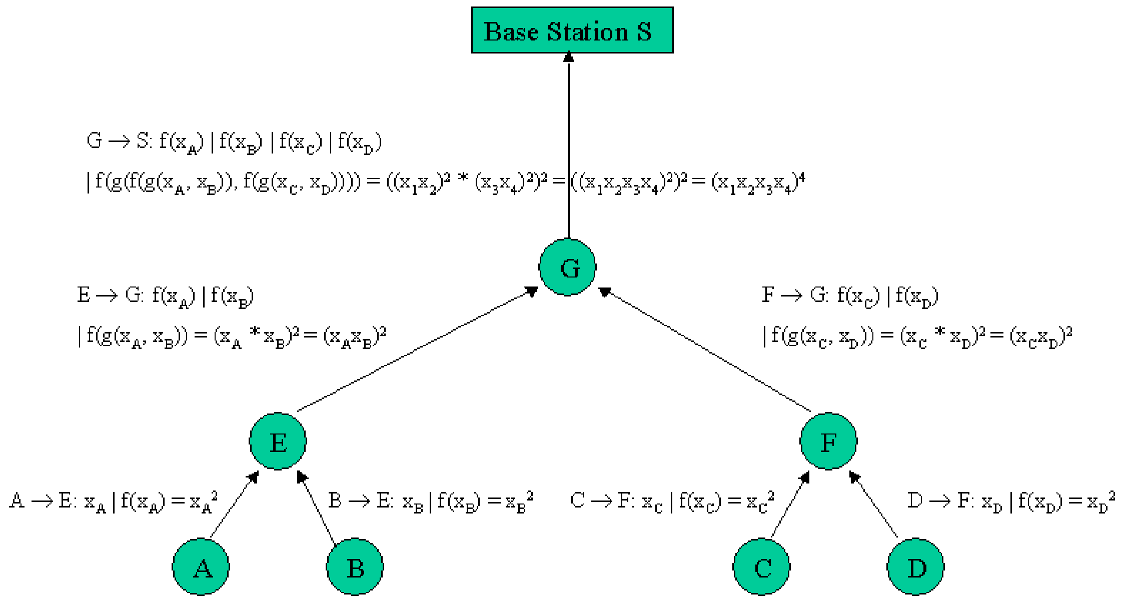

Base Station S
$G \rightarrow S$: $f(x_A) | f(x_B) | f(x_C) | f(x_D)$
$| f(g(f(g(x_A, x_B)), f(g(x_C, x_D)))) = ((x_1x_2)^2 * (x_3x_4)^2)^2 = ((x_1x_2x_3x_4)^2)^2 = (x_1x_2x_3x_4)^4$
G
$E \rightarrow G$: $f(x_A) | f(x_B)$
$| f(g(x_A, x_B)) = (x_A * x_B)^2 = (x_Ax_B)^2$
$F \rightarrow G$: $f(x_C) | f(x_D)$
$| f(g(x_C, x_D)) = (x_C * x_D)^2 = (x_Cx_D)^2$
E
F
$A \rightarrow E$: $x_A | f(x_A) = x_A^2$
$B \rightarrow E$: $x_B | f(x_B) = x_B^2$
$C \rightarrow F$: $x_C | f(x_C) = x_C^2$
$D \rightarrow F$: $x_D | f(x_D) = x_D^2$
A
B
C
D

Figure 8 Generation of Signed Aggregate

Intermediate sensors calculate the aggregate, g(x, y), sign the aggregate, f(g(x, y)) and transmit the signatures received from their children along with the signed aggregate. The base station, responsible for verifying the data, aggregates the signatures received from the leaf sensors, two at a time, signs the aggregated signatures and aggregates the signed aggregated signatures, Figure 9b. The base station then compares the signed aggregates received from the network to the aggregated signatures calculated, Figure 9c. If the two match, it is concluded that the data was not tampered with and can be trusted.

Base Station Receives:

(a) $f(x_A) = x_A^2,\ f(x_B) = x_B^2,\ f(x_C) = x_C^2,\ f(x_D) = x_D^2,\ f(g(f(g(x_A, x_B)), f(g(x_C, x_D)))) = (x_A x_B x_C x_D)^4$

Base Station Calculates:

$g(f(x_A), f(x_B)) = (x_A^2) * (x_B^2) = (x_A x_B)^2$

$g(f(x_C), f(x_D)) = (x_C^2) * (x_D^2) = (x_C x_D)^2$

(b) $f(g(f(x_A), f(x_B))) = ((x_A x_B)^2)^2 = (x_A x_B)^4$

$f(g(f(x_C), f(x_D))) = ((x_C x_D)^2)^2 = (x_C x_D)^4$

$g(f(g(f(x_A), f(x_B))), f(g(f(x_C), f(x_D)))) = (x_A x_B)^4 * (x_C x_D)^4 = (x_A x_B x_C x_D)^4$

Base Station Verifies:

(c) $f(g(f(g(x_A, x_B)), f(g(x_C, x_D)))) = g(f(g(f(x_A), f(x_B))), f(g(f(x_C), f(x_D))))$

$(x_A x_B x_C x_D)^4 = (x_A x_B x_C x_D)^4$

Figure 9 Verification Process

Though the idea proved to be possible, it turned out to not be secure. The exponent used in the signature function was the same for every computation of a signature thus implying that all sensors used the same key. To correct this, the exponent for every signature must be unique so that each sensor would use its secret key to sign the data, as in Figure 10.

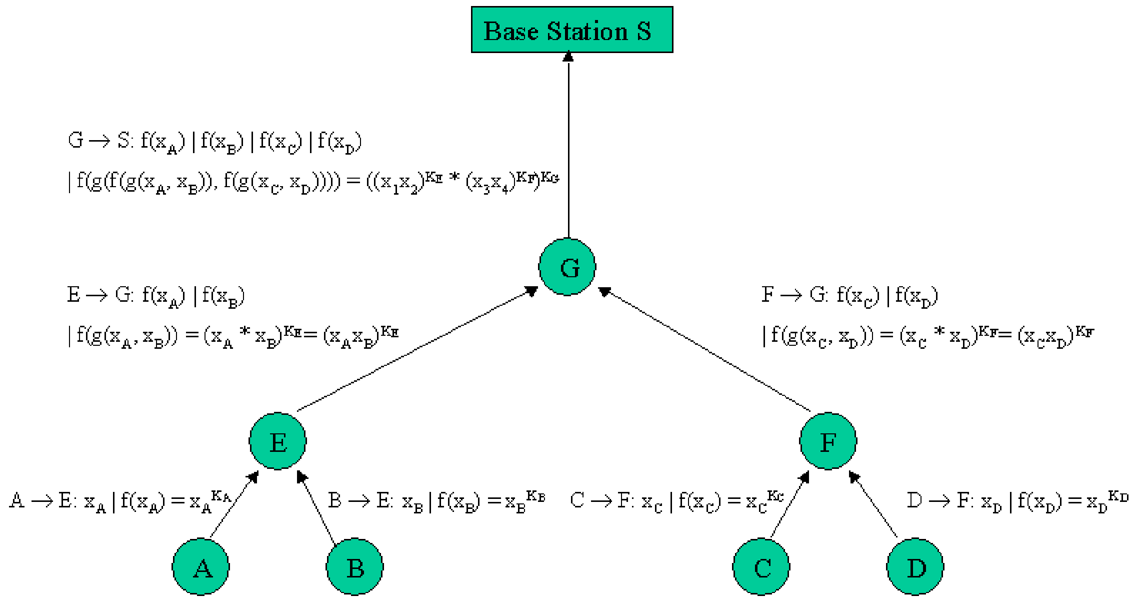

Base Station S
G → S: f(x_A) | f(x_B) | f(x_C) | f(x_D)
| f(g(f(g(x_A, x_B)), f(g(x_C, x_D)))) = ((x_1x_2)^KE * (x_3x_4)^KF)^KG
G
E → G: f(x_A) | f(x_B)
| f(g(x_A, x_B)) = (x_A * x_B)^KE = (x_Ax_B)^KE
F → G: f(x_C) | f(x_D)
| f(g(x_C, x_D)) = (x_C * x_D)^KF = (x_Cx_D)^KF
E
F
A → E: x_A | f(x_A) = x_A^KA
B → E: x_B | f(x_B) = x_B^KB
C → F: x_C | f(x_C) = x_C^KC
D → F: x_D | f(x_D) = x_D^KD
A
B
C
D

Figure 10 Secure Generation of Signed Aggregate

Unfortunately, no matter what keys we use, this process cannot work, as the distributive property does not hold. This can be seen in Figure 11c. We were unable to derive a composite function approach that guaranteed data integrity and collusion protection.

Base Station Receives:

(a) $f(x_A) = x_A^{K_A}, f(x_B) = x_B^{K_B}, f(x_C) = x_C^{K_C}, f(x_D) = x_D^{K_D}, f(g(f(g(x_A, x_B)), f(g(x_C, x_D)))) = ((x_1 x_2)^{K_E} * (x_3 x_4)^{K_F})^{K_G}$

Base Station Calculates:

$g(f(x_A), f(x_B)) = (x_A^{K_A}) * (x_B^{K_B}) = x_A^{K_A} x_B^{K_B}$

$g(f(x_C), f(x_D)) = (x_C^{K_C}) * (x_D^{K_D}) = x_C^{K_C} x_D^{K_D}$

(b)

$f(g(f(x_A), f(x_B))) = (x_A^{K_A} x_B^{K_B})^{??}$

$f(g(f(x_C), f(x_D))) = (x_C^{K_C} x_D^{K_D})^{??}$

$g(f(g(f(x_A), f(x_B))), f(g(f(x_C), f(x_D)))) = (x_A^{K_A} x_B^{K_B})^{??} * (x_C^{K_C} x_D^{K_D})^{??}$

Base Station Verifies:

(c) $f(g(f(g(x_A, x_B)), f(g(x_C, x_D)))) = g(f(g(f(x_A), f(x_B))), f(g(f(x_C), f(x_D))))$

$$((x_1 x_2)^{K_E} * (x_3 x_4)^{K_F})^{K_G} \neq (x_A^{K_A} x_B^{K_B})^{??} * (x_C^{K_C} x_D^{K_D})^{??}$$

Figure 11 Secure Verification Process

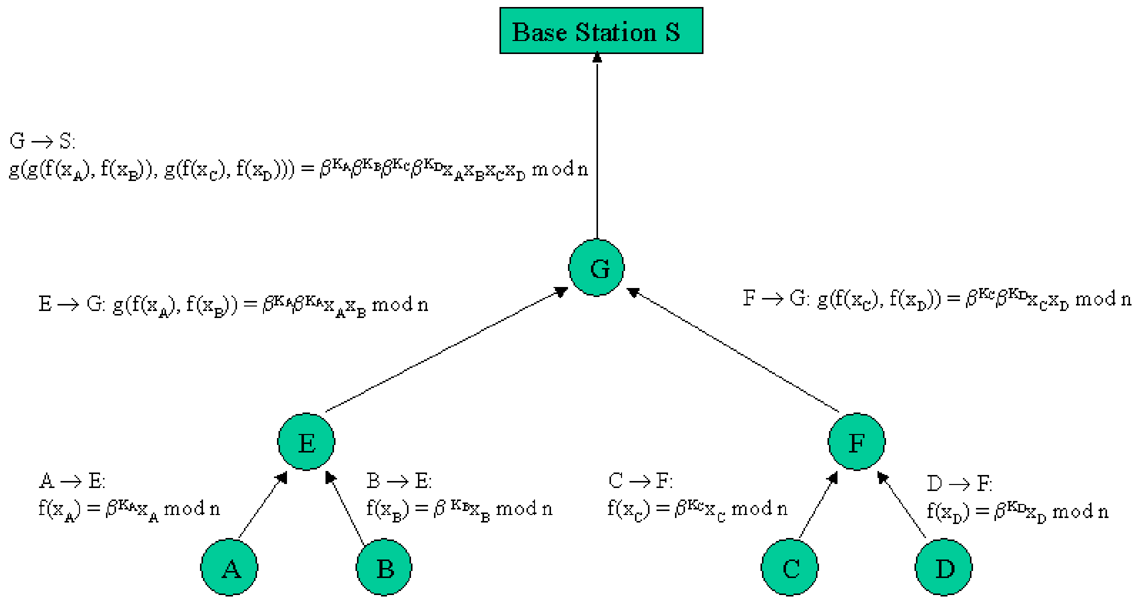
We turned our efforts towards trying to find two different functions that would provide the level of data integrity and collusion prevention we were looking for. Once we found the functions, we analyzed them and determined the properties necessary to make it work. We tried different compositions using RSA, ElGamal, multiplication and exponentiation. Using the ElGamal encryption scheme in conjunction with multiplication allowed us to extract what appeared to be an aggregate. Refer to section 5.2.1. for a more detailed explanation.

5.2. Composing Encryption and Aggregation Functions

The purpose of privacy preserving data aggregation is to encrypt and aggregate the data in such a way that intermediate sensors do not need to decrypt the data in order to perform the aggregation and that the final aggregated value can be decrypted in the end by the base station. The distributive nature of the aggregation and encryption functions allows these features.

The composition of an encryption function and aggregation function that we were looking for was aggregate(encrypt(x), encrypt(y)). We then wanted to apply the decryption function to the encrypted aggregate to extract the aggregate value. We wanted decrypt(aggregate(encrypt(x), encrypt(y))) = aggregate(x, y) to hold true.

Composing an ElGamal-like encryption function, $f(x) = \beta^{K_A}x \bmod n$, with a multiplication aggregation function, $g(x, y) = xy$, allowed us to aggregate encrypted data without first decrypting, Figure 12.



Base Station S
G → S:
g(g(f(x_A), f(x_B)), g(f(x_C), f(x_D))) = β^{K_A}β^{K_B}β^{K_C}β^{K_D}x_Ax_Bx_Cx_D mod n
E → G: g(f(x_A), f(x_B)) = β^{K_A}β^{K_A}x_Ax_B mod n
F → G: g(f(x_C), f(x_D)) = β^{K_C}β^{K_D}x_Cx_D mod n
G
E
F
A → E: f(x_A) = β^{K_A}x_A mod n
B → E: f(x_B) = β^{K_B}x_B mod n
C → F: f(x_C) = β^{K_C}x_C mod n
D → F: f(x_D) = β^{K_D}x_D mod n
A
B
C
D

Figure 12 Generation of Encrypted Aggregate

To get the aggregate, we multiplied the final encrypted aggregate by β^{-K_A}, where $-K_A$ is the decryption key:

$$\text{Decrypt: } (\beta^{K_A}\beta^{K_B}\beta^{K_C}\beta^{K_D}x_Ax_Bx_Cx_D \bmod n))\beta^{-K_A}\beta^{-K_B}\beta^{-K_C}\beta^{-K_D} = x_Ax_Bx_Cx_D \bmod n \qquad (2)$$

Thus, the relationship we seek holds and our desired equality exists.

Clearly, there are limitations to this solution. First, this specific solution is only applicable when the aggregation function is multiplication. This, in fact, may not be as limiting as it first appears. The authors of [2] assume that the aggregation function is distributive, which multiplication is. Second, and of less concern, is that the decrypted aggregate is calculated mod n. This means that there is a possibility of not obtaining a true aggregate. To ensure that the result of the decryption is the true aggregate, the modulus n must be larger than the largest possible value in the set of aggregate values.

Nonetheless, this construction illustrates the properties that we seek: 1) privacy preservation and 2) collusion resistance.

5.3. Application of PPDA

In this section, we show how using PPDA with concepts from SPINS and the Hu-Evans protocol satisfies the four security requirements. PPDA provides data confidentiality while data integrity, authentication and freshness are achieved using concepts from SPINS and the Hu-Evans protocol.

5.3.1. Assumptions

Some of the architecture, cryptographic and trust assumptions made in [2] and [3] are applied in this thesis:

- The base station is much more powerful than the sensor devices. It has more power and memory and is equipped to handle expensive computations.
- The base station is completely trusted by the sensor devices and the observer. The base station is a single point of failure so this assumption is necessary to keep the network from being useless.
- Sensor devices cannot be trusted by the base station and observer and thus must be protected against.

- Sensor devices must trust themselves. This is necessary for the purpose of time and counter synchronization. It must be believed that the internal clock of the sensor is accurate.
- It is essential that each sensor device is able to establish shared secrets with the base station before they are deployed.
- Cryptographic algorithms used to generate keys and MACs must be efficient and secure. No particular encryption or MAC algorithm is required.
- A simple hierarchical tree aggregation protocol rooted at the base station is assumed. Data flows from the leaf sensors towards the base station as it is aggregated.
- Upon deployment, the sensors use a secure, self-organizing protocol to form a hierarchical tree routing path through which data will be aggregated and transmitted.
- Data delivery is guaranteed through some predetermined network routing model. In other words, it is assumed that no data is lost in transmission.
- The aggregation function must be distributive and it must not matter what order sensor data is aggregated [2].

In addition, we have made the following assumptions:

- The sensor network is deployed in a hostile environment where sensitive data is being transmitted by the sensor devices so confidentiality is a necessity with respect to sensor data.
- The base station is not broadcasting sensitive data so confidentiality is not a necessity with respect to data broadcast by the base station. However, data transmitted directly to a sensor may be sensitive and needs to be protected.
- The base station and sensor devices are static. The event being observed may or may not be mobile.

- The protocol described in this thesis does not depend on the data delivery model used. No particular delivery model is assumed.

5.3.2. Notation

Some of the notation used in our protocol stems from that of [2] and [3]:

A – H	Sensor devices
S	Base station
$A \rightarrow B$	Sensor A sends to sensor B; denotes data transmission
ID_A	Unique identifier of sensor A
$M_1 \mid M_2$	Concatenation of messages M_1 and M_2
E(K, M)	Encryption of message M with key K
MAC(K, M)	Generation of the MAC on message M using key K
Agg(x, y)	Aggregation of x and y
i	time interval
x_A	Data collected by sensor A
K_i	i^{th} key of the base station broadcast key chain
K_{AS}	Shared symmetric key used to generate temporary MAC keys and used for direct communication between base station and sensor
K_{AS}’	Shared symmetric key between base station and sensor used to generate temporary encryption keys
K_{SA}’	Encryption key used by base station for direct communication between base station and sensor
K_{Ai}	i^{th} key of the set of temporary MAC keys of sensor A
K_{Ai}’	i^{th} key of the set of temporary encryption keys of sensor A

5.3.3. Layout of Sensor Network

The sensor network used in this thesis is a randomly distributed, static ad hoc network connected to the observer’s outside network through the base station (S). Once the network has been deployed and configured, the leaf sensors (A, B, C, D, E, F, G and H) begin collecting data. The data is transmitted to the base station using a hop-by-hop routing protocol. Intermediate sensors (E, F, G and H) aggregate the data using delayed data aggregation. Leaf sensors (A, B, C and D) are not responsible for aggregating data.

5.3.4. Key Generation

The keys used in our protocol can be divided into five categories: base station broadcast keys, K_i, base station encryption keys, K_{SA}, shared symmetric keys, K_{AS} and

K_{AS}’, temporary MAC keys, K_{Ai}, and temporary encryption keys, K_{Ai}’. The base station broadcast keys are a set of keys generated by the base station to use whenever it is necessary to broadcast data to the sensors. This set of keys is used to compute MACs. It is the same as the key chain of keys generated in the μTESLA protocol and so is generated the same way, refer back to Figure 1a.

The base station encryption keys, K_{SA}, are used for direct communication between the base station and sensor whenever the base station needs to transmit sensitive data to a specific sensor. For n sensors in the network, the base station stores n encryption keys.

The symmetric keys, K_{AS} and K_{AS}’, are shared between the base station and each sensor device and are used to generate the temporary MAC keys and the temporary encryption keys. Each sensor stores 2 shared symmetric keys and, for n sensors in the network, the base station stores 2n shared symmetric keys. To generate the temporary MAC keys, key K_{AS} is used to encrypt a counter value[6]. To generate the temporary encryption keys, key K_{AS}’ is used to encrypt the same counter value. In addition, keys K_{AS} are used by the base station to generate MACs on the data that the base station needs to transmit to a specific sensor.

As a requirement, the base station and sensor clocks are synchronized so that the time intervals and counter values are in sync. This, along with the fact that the key used to generate the temporary encryption and MAC keys is shared by the base station and sensor, allows the base station to calculate the temporary encryption and MAC keys without needing the sensors to transmit any additional data, such as the counter value.

Prior to deployment, sensors are initialized with three keys: the first key in the key chain generated by the base station, K_0, and the two symmetric keys shared with the base station, K_{AS} and K_{AS}’. The temporary MAC and encryption keys are generated by the sensors at the beginning of each time interval.

5.3.5. Data Communication Between Sensors and Base Station

We define three communication categories, that exist in the sensor network used in this thesis: 1) base station to all sensor devices, 2) base station to one sensor device and

[6] For all intents and purposes, this counter value is the same as the time interval, i.

3) sensor device to base station. The first category is used whenever the base station needs to broadcast queries or data, such as keys, to every sensor in the network. In this category, the base station uses the key chain of MAC keys to generate MACs on data broadcast to all sensors.

The second category is used whenever the base station needs to transmit a query, a request or a piece of data to a specific sensor. In this category, the base station uses key K_{AS} to generate MACs on data transmitted directly to the sensor and key K_{SA} to encrypt data transmitted directly to the sensor.

The third category is used when the sensors need to transmit collected data to the base station. In this category, sensors use key K_{Ai} to generate MACs on data transmitted to the base station and key K_{Ai}' to encrypt data transmitted to the base station.

5.3.6. Data Authentication, Encryption and Aggregation

Using the delayed authentication technique of µTESLA, sensors store data received and wait to authenticate it until the MAC key has been revealed[7] by the base station. Once the base station reveals the key, the sensors use it to calculate MACs on the data they received and compare them to the MACS they received. If they match, then it is concluded that the data is authentic.

Using PPDA, sensors encrypt their data with their secret encryption key and forward this to their parents without aggregating their encrypted data with the encrypted data received from their grandchildren. Sensors are only responsible for aggregating their grandchildren's encrypted data. The base station decrypts the encrypted aggregate using the secret decryption keys of the sensors.

5.3.7. A Simple Example

In this example, the protocol is described where the encryption function is $E(K_{Ai}', x_A) = \beta^{K_{Ai}'} x_A \bmod n$ and the aggregation function is $Agg(x, y) = xy$. Also, the leaf sensors are the only sensors collecting data. For this reason, encryption is performed

[7] Intermediate sensors are not required to decrypt data before performing aggregation. For this reason, it is not necessary to reveal temporary encryption keys. Only the temporary MAC keys are revealed.

only by the leaf sensors. In a network where all sensors are collecting data, each sensor would be required to perform encryption on the data it is transmitting.

Leaf sensors A, B, C and D are responsible for transmitting their unique identifiers, encrypted data collected and MACs on the encrypted data:

$$\begin{aligned}
A \rightarrow E: &\quad ID_A \mid E(K_{Ai}', x_A) \mid MAC(K_{Ai}, E(K_{Ai}', x_A)) \\
B \rightarrow E: &\quad ID_B \mid E(K_{Bi}', x_B) \mid MAC(K_{Bi}, E(K_{Bi}', x_B)) \\
C \rightarrow F: &\quad ID_C \mid E(K_{Ci}', x_C) \mid MAC(K_{Ci}, E(K_{Ci}', x_C)) \\
D \rightarrow F: &\quad ID_D \mid E(K_{Di}', x_D) \mid MAC(K_{Di}, E(K_{Di}', x_D))
\end{aligned}$$

Parents of leaf sensors, E and F, are responsible for forwarding the data received from each of their children, along with their unique identifiers and a MAC generated on the aggregate of the encrypted data received from their children.

$$\begin{aligned}
E \rightarrow G: &\quad ID_A \mid E(K_{Ai}', x_A) \mid MAC(K_{Ai}, E(K_{Ai}', x_A)) \\
&\quad \mid ID_B \mid E(K_{Bi}', x_B) \mid MAC(K_{Bi}, E(K_{Bi}', x_B)) \\
&\quad \mid ID_E \mid MAC(K_{Ei}, Agg(E(K_{Ai}', x_A), E(K_{Bi}', x_B))) \\
F \rightarrow G: &\quad ID_C \mid E(K_{Ci}', x_C) \mid MAC(K_{Ci}, E(K_{Ci}', x_C)) \\
&\quad \mid ID_D \mid E(K_{Di}', x_D) \mid MAC(K_{Di}, E(K_{Di}', x_D)) \\
&\quad \mid ID_F \mid MAC(K_{Fi}, Agg(E(K_{Ci}', x_C), E(K_{Di}', x_D)))
\end{aligned}$$

Every intermediate sensor from the level above the parents of leaf sensors up to the top-most level, is responsible for calculating and transmitting the aggregate on the encrypted data received from their grandchildren, along with the identifiers of their children, the MACs generated by their children on the encrypted data received from their grandchildren and a MAC generated on the aggregate of the aggregates received from their children. For example, intermediate sensor G would calculate and transmit the aggregate on the encrypted data received from its grandchildren (A, B, C and D), along with the identifiers of its children (E and F), the MACs generated by its children (E and

F) on the encrypted data received from its grandchildren (A, B, C and D) and a MAC generated on the aggregate of the aggregates received from its children (E and F).

$$\begin{array}{ll}
G \rightarrow H: & ID_E \mid Agg(E(K_{Ai}', x_A), E(K_{Bi}', x_B)) \mid MAC(K_{Ei'}, Agg(E(K_{Ai}', x_A), E(K_{Bi}', x_B))) \\
 & \mid ID_F \mid Agg(E(K_{Ci}', x_C), E(K_{Di}', x_D)) \;\; MAC(K_{Fi'}, Agg(E(K_{Ci}', x_C), E(K_{Di}', x_D))) \\
 & \mid ID_G \mid MAC(K_{Gi'}, Agg((Agg(E(K_{Ai}', x_A), E(K_{Bi}', x_B)), (Agg(E(K_{Ci}', x_C), E(K_{Di}', x_D))))) \\
H \rightarrow S: & ID_G \mid ID_H \\
 & \mid Agg((Agg(E(K_{Ai}', x_A), E(K_{Bi}', x_B)), (Agg(E(K_{Ci}', x_C), E(K_{Di}', x_D))))) \\
 & \mid MAC(K_{Gi'}, Agg((Agg(E(K_{Ai}', x_A), E(K_{Bi}', x_B)), (Agg(E(K_{Ci}', x_C), E(K_{Di}', x_D)))))
\end{array}$$

Upon receipt of the final encrypted aggregate, $\beta^{K_{Ai}'}\beta^{K_{Bi}'}\beta^{K_{Ci}'}\beta^{K_{Di}'}x_A x_B x_C x_D \bmod n$, the base station decrypts it to obtain the final aggregated data.

Decrypt: $(\beta^{K_{Ai}'}\beta^{K_{Bi}'}\beta^{K_{Ci}'}\beta^{K_{Di}'}x_A x_B x_C x_D \bmod n)(\beta^{-K_{Ai}'}\beta^{-K_{Bi}'}\beta^{-K_{Ci}'}\beta^{-K_{Di}'}) = x_A x_B x_C x_D \bmod n$

This illustration shows the process that allows sensors to aggregate encrypted data without decrypting the data beforehand.

CHAPTER 6
CONCLUSION

In this thesis, we provide a method of aggregating encrypted data without needing to decrypt it first. Our technique, privacy preserving data aggregation (PPDA), uses the idea of composition of functions and the distributive property of the selected aggregation function over the selected encryption function to provide protection against data leakage and collusion attacks. By composing an aggregation function and an encryption function, we are able to protect the confidentiality of the data in such a way that two or more sensors are unable to collude to understand and possibly leak sensitive data and still perform aggregation.

In addition, applying PPDA to two security protocols, the Hu-Evans protocol and the μTESLA protocol, fulfills the four security requirements: data integrity, data authentication, data confidentiality and data freshness. Data integrity is ensured via delayed data aggregation in the Hu-Evans protocol. Data authentication is guaranteed via delayed data authentication in the μTESLA protocol and Hu-Evans protocol. Data confidentiality is obtained via composition of distributive functions in PPDA. Data freshness is achieved via counters and keys used during specific time intervals.

Although we were unable to derive a composite function approach that guaranteed data integrity, we view PPDA as the beginning of finding an integrity-ensuring aggregation mechanism.

6.1. Future Work

Future work may include finding the appropriate and necessary properties of and relationship(s) between two functions, an aggregation function and possibly a signature function, such that the composition of the two allows data integrity to be ensured. Finding these properties and understanding these relationships will provide more insight into the types of functions necessary for this aggregation mechanism to work. It would be ideal to find two functions such that Figure 7b holds true. It is our belief that this solution would provide the integrity guarantees and the collusion prevention we are looking for.

REFERENCES

[1] Hasan Cam, Suat Ozdemir, Devasenapathy Muthuavinashiappan and Prashant Nair. Energy-Efficient Security Protocol for Wireless Sensor Networks. IEEE VTC. October 2003.

[2] Lingxuan Hu and David Evans. Secure Aggregation for Wireless Networks. Workshop on Security and Assurance in Ad-Hoc Networks. January 2003.

[3] Adrian Perrig, Robert Szewczyk, J. D. Tygar, Victor Wen and David Culler. SPINS: Security Protocols for Sensor Networks. Wireless Networks Journal (WINE). September 2002.

[4] Ian F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam and Erdal Cayirci. A Survey on Sensor Networks. IEEE Communications Magazine. August 2002.

[5] Konstantinos Kalpakis, Koustuv Dasgupta and Parag Namjoshi. Maximum Lifetime Data Gathering and Aggregation in Wireless Sensor Networks. Proceedings of the 2002 IEEE International Conference on Networking. August 2002.

[6] Bhaskar Krishnamachari, Deborah Estrin and Stephen Wicker. The Impact of Data Aggregation in Wireless Sensor Networks. International Workshop on Distributed Event-Based Systems. July 2002.

[7] Chalermek Intanagonwiwat, Deborah Estrin, Ramesh Govindan and John Heidemann. Impact of Network Density on Data Aggregation in Wireless Sensor Networks. International Workshop on Distributed Event-Based Systems. July 2002.

[8] Sameer Tilak, Nael B. Abu-Ghazaleh and Wendi Heinzelman. A Taxonomy of Wireless Micro-Sensor Network Models. ACM Mobile Computing and Communication Review. April 2002.

[9] K. Sohrabi, J. Gao, V. Ailawadhi and G. J. Pottie. Protocols for Self-Organization of a Wireless Sensor Network. IEEE Personal Communications. October 2000.

[10] Chalermek Intanagonwiwat, Ramesh Govindan and Deborah Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. Mobile Computing and Networking. August, 2000.

[11] William A. Wulf, Alec Yasinsac, Katie S. Oliver and Ramesh Peri. A Technique for Remote Authentication. Proceedings of the Internet Society Symposium on Network and Distributed System Security. 1994.

[12] Bartosz Przydatek, Dawn Song and Adrian Perrig. SIA: Secure Information Aggregation in Sensor Networks. Conference on Embedded Networked Sensor Systems. November 2003.

[13] H. Cam, S. Ozdemir, P. Nair and D. Muthuavinashiappan. ESPDA: Energy-Efficient and Secure Pattern-Based Data Aggregation for Wireless Sensor Networks. IEEE Sensors 2003 Conference. October 2003.

[14] Samuel Madden, Michael J. Franklin and Joseph M. Hellerstein. TAG: a Tiny Aggregation Service for Ad-Hoc Sensor Networks. OSDI. December 2002.

BIOGRAPHICAL SKETCH

Khandys A. Polite graduated with a Bachelor of Science degree in Computer Science from The Florida State University in December 1998. Upon graduation, she began working for the Division of Emergency Management (DEM) in the Department of Community Affairs (DCA) in Tallahassee, Florida. She returned to school in September 2002 to pursue a Master of Science degree in Computer Science with a specialization in Information Security. After receiving her Master of Science degree, she will begin working for the Naval Research Laboratory in Washington, D.C.

Khandys is a member of the Upsilon Pi Epsilon (UPE) Computing Sciences Honor Society, the Association for Computing Machinery (ACM) and the Golden Key National Honor Society. She participated in the Department of Defense (DoD) Information Assurance Scholarship Program (IASP), the Office of Naval Research (ONR) Internship Program and the Minority Scholars Program (MSP).